Probability estimation and a competence model for rule based e-tutoring systems

Diederik M. Roijers

Johan Jeuring

Ad Feelders

Technical Report UU-CS-2012-003
March 2012

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

Probability estimation and a competence model for rule based e-tutoring systems

Diederik M. Roijers, Johan Jeuring, Ad Feelders
Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands

ABSTRACT

In this paper, we present a student model for rule based e-tutoring systems. This model describes both properties of rewrite rules (difficulty and discriminativity) and of students (start competence and learning speed). The model is an extension of the two-parameter logistic ogive function of Item Response Theory. We show that the model can be applied even to relatively small datasets. We gather data from students working on problems in the logic domain, and show that the model estimates of rule difficulty correspond well to expert opinions. We also show that the estimated start competence corresponds well to our expectations based on the previous experience of the students in the logic domain. We point out that this model can be used to inform students about their competence and learning, and teachers about the students and the difficulty and discriminativity of the rules.

Categories and Subject Descriptors
J.1 [Administrative Data Processing] Education; K.3.1 [Computer Uses in Education] Collaborative learning, Computer-assisted instruction (CAI), Computer-managed instruction (CMI), Distance learning

Keywords
Learning Analytics, Student Model, Data Mining

1. INTRODUCTION

Students of natural sciences learn to solve various types of standard problems. Many of these problems are solved by rewriting some kind of formula or expression, step by step, until the problem is solved. Each problem domain (e.g. algebra, matrix calculus, or logic) has its own set of rewrite rules. Students start practicing these types of problems early in their school careers: already at primary school they learn how to calculate with fractions.

Learning to solve problems using rewriting is often a time-consuming and labor-intensive process, both for students and teachers. Students have to practice a lot, and often require a lot of feedback, which a teacher may not always be able to give immediately. E-tutoring systems can be used to alleviate this situation by providing feedback automatically. In recent years the Open University the Netherlands has developed an e-tutoring system based on rules and strategies: the `ideas` framework[1]. Heeren et al. [3] show how to use rewrite rules and strategies in e-tutoring systems to provide feedback. Feedback can be provided for every (correct or incorrect) application of a rewrite rule, but not about how difficult and discriminative rules are, how competent students are when they start working, and how fast they learn.

In this paper we answer the question: *How can we describe student behavior in a rule based e-tutoring system with a student model, and estimate the probability of a student applying a rule correctly, the next time he/she tries to apply it?* The model describes the behavior of the students (whether or not a student applies the rewrite rules correctly) in relation to the rules. Therefore, it includes the properties of a rewrite rule, as well as properties of a student.

Previous research on student models in the context of e-tutoring systems suggests that Item Response Theory (IRT) is a good starting point. Johns et al. [4] apply IRT to an e-tutoring system that offers multiple choice questions for mathematics tests. However, multiple choice questions are single static questions, while we want to model recurring rewrite rules. We therefore have to adjust the model to incorporate rewrite rules, and to allow a student to apply the same rule multiple times.

Cen et al. [2]present a student model based on logistic regression, as we will do, but with more parameters than we use. They use a simplifying assumption that students learn at the same rate, which is of course invalid in general.

This paper is organised as follows. Section 2 discusses the method we use in our research. Section 3 presents our student model, and shows the results of testing this model by simulation and using real life data. Section 4 concludes and discusses what needs to be done to further evaluate this model, and how the model can be used in practice.

2. METHODS

This research consists of two phases: a construction phase, in which we create a student model and test it on simulated

[1] `http://ideas.cs.uu.nl/`

LAK '12, 29 April – 2 May 2012, Vancouver, BC, Canada

data, and a validation phase in which we use real-life data from students working on an e-tutoring system to learn the model parameters and compare these learned parameters to expert opinions.

To create a student model for rule based e-tutoring systems we use the IRT framework, and more specifically the two parameter logistic ogive function (2PL), as a basis. One reason why we select 2PL, is because it is a well-established model in psychology and education (e.g. used to validate tests). An even more significant reason is that it operationalizes the desired properties both for rules (items in IRT) and student: difficulty and discriminativity, and competence. It relates these properties to the probability of a correct application of a rule (a positive response in IRT) [1].

We have investigated four alternative models:

1. The start competence per student per rule may be different, as well as the learning speed.
2. The start competence per rule may be different, but the learning speed is always the same for one student.
3. The start competence is the same for all rules for a student, but the learning speed may vary.
4. The start competence and the learning speed do not depend on the rule, but are properties of a student only.

Our domain experts discard option 3, because the learning speed is more likely to be a constant parameter over all rules for a single student than the start competence. We test the other options using simulated data. We run several simulations to discover how the model and the learning algorithms we use (see next section) behave in terms of: approximating the true parameter value, variance in the estimated parameters, sensitivity to the amount of data (number of rules, number of students, number of instances per rule per student), and the accuracy with which the outcomes (applying a rule either correctly or incorrectly) are predicted.

For validating the model we collect data from students using a rule- and strategy-based e-tutoring system to learn to rewrite boolean expressions into disjunctive normal form (DNF). This subject is often taught in Computer Science programmes, and we have a number of domain experts available at the two universities where we collect data. We collect data at Utrecht University (UU) and Utrecht University of Applied Sciences (HU). We ask the domain experts to rank the rewrite rules necessary for the task twice: once for difficulty and once for discriminativity. We compare these rankings to the rankings based on the estimated values for the parameters from the student data returned by our learning algorithm. The rewrite rules for this domain can be found in Table 1.

We also compare what we know about the previous knowledge of the students to the output values of the starting competence parameter of the student model. There are three groups of student participants: 4 UU students with recent training in solving DNF problems, which we presume to have most previous knowledge; 5 HU second-year technical computer science students, who have had less training in manipulating logic expressions, but do have one and a half years of programming experience; and 5 HU first-year business informatics students, who have had little training in manipulating logical expressions and little programming experience, who we assume to have least previous knowledge.

3. RESULTS

The results section is divided into two subsections: construction and validation. In the construction subsection we describe how the model is constructed, and how it performs on simulated data. In the validation subsection we show the results of applying the model to real data.[2]

3.1 Construction

As a starting point for model construction we choose the 2PL IRT model. 2PL relates the probability of a correct answer for a number of items (e.g. problems on a test) for a number of students, to the latent variables of student competence and item difficulty and discriminativity, through the following probability function:

$$P(o_{i,j} = 1|\theta_i, b_j, a_j) = \frac{e^{a_j(\theta_i - b_j)}}{1 + e^{a_j(\theta_i - b_j)}} \quad (1)$$

where the binary outcome $o_{i,j}$ is the outcome for student i attempting item j, a_j and b_j are the discriminativity and difficulty of item j, and θ_i is the competence of student i. The data is a binary matrix, containing the outcomes for each student for each exercise. The parameter values can be learned through an iterative scheme, alternating between optimizing the likelihood by changing a_j and b_j, and optimizing the likelihood by changing θ_i, using gradient descent [1]. The 2PL model assumes that the competence is constant for each student.

The data we gather from an e-tutoring system however, is different from a binary matrix. A student does several exercises in which each rule can be attempted several times, with the purpose to increase his or her competence. We obtain data in the form of a sequence of tuples of student ID, rule ID and the outcome (0 or 1). We therefore know how many times a student has attempted a rule before when (s)he applies a rule.

We extend 2PL, by adding a learning parameter. The current competence for a rule for a student is now the start competence plus the learning parameter times the number of times the rule has been attempted before. As described in the methods section we have to choose whether the starting competence and learning speed are parameters of the student, or whether they can also vary per rule. After testing the different models, we conclude that the variance in the learned parameters is too high to be used for prediction or to be informative to teachers and students when they depended on both the student and the rule. We therefore let the starting competence and the learning parameters be attributes of

[2]More details can be found in the Master's thesis of the first author available via `http://www.staff.science.uu.nl/~jeuri101/homepage/Publications/ThesisDMRoijers.pdf`

a student only. The following probability function describes our model:

$$P(o_{r,s,t} = 1) = \frac{e^{a_r(\theta_{0,s}+\eta_s t_{r,s}-b_r)}}{1 + e^{a_r(\theta_{0,s}+\eta_s t_{r,s}-b_r)}} \quad (2)$$

where r is the rule ID, s the student ID, $\theta_{0,s}$ is the starting competence of the student, η_s the learning speed of the student, and $t_{r,s}$ the number of times student s attempted rule r before.

We run several simulations using this model. For n students, the starting competence $\theta_{0,s}$ is chosen randomly from a uniform distribution between -3 and 1, and the learning parameter η_s is chosen randomly from a uniform distribution between 0 and 0.5. For m rules the parameter values are also drawn from uniform distributions, for a_r between 0.8 and 1.5 and for b_j between -3 and 3. We choose a value t_{max} for the number of outcomes per student per rule. Using the randomly chosen rule and student parameter values we simulate data, by drawing outcomes stochastically, with the probability of a positive outcome given by equation (2). Using the simulated data we learn the parameters back: we generate parameter values, with these generated parameters as the original parameters we generate outcomes, and (ignoring the original parameters) we estimate the parameters using the data. This process we call parameter recovery. When recovering the parameters for small data sets we cannot use gradient descent. Gradient descent cannot be applied when there is linear separability of the data, or when there are either no positive or no negative outcomes for a rule. The probability that one of these issues occurs is high when the dataset is small. We therefore replace gradient descent by attempting a discrete set of parameter values (with intervals of 0.01) and determine which combination of item and student parameters yields the highest likelihood.

When we use parameter recovery simulations we find that our model and learning algorithms are unbiased by calculating the average difference between the original randomly chosen parameter values, and the recovered parameter values (which should be close to 0). We can also show that the learning is reliable by calculating the mean absolute difference between the original and the recovered parameter values. For a simulation with 50 students, 25 rules, and 20 instances per rule, the average difference between the recovered and the original parameter values is around 1% of the parameter value range. The mean absolute differences between the original and recovered parameters are: 0.015 for η_s, 0.15 for $\theta_{0,s}$, 0.07 for a_r, and 0.09 for b_r. These "errors" are acceptable.

The simulation closest to the real data obtained is 15 students, 23 rules, and 8 instances per rule. Here the mean absolute differences between the original and recovered parameters are: 0.09 for η_s, 0.38 for $\theta_{0,s}$, 0.21 for a_r, and 0.34 for b_r. For $\theta_{0,s}$ and b_r, these errors are still acceptable, but for η_s it is almost one fifth of the parameter range, and for a_r it is two fifth of the range. We therefore conclude that the parameter value estimates of η_s and a_r are unreliable for this data size.

The average accuracy of predicting the outcomes is determined as follows. We have original and recovered parameter values. We use the original rule parameters and new randomly drawn student parameters to generate more data. The data comes in as a stream of outcomes. At each point in time a random rule is selected, a prediction is made whether this outcome will be positive or negative, and then an outcome is generated with the right probabilities (based on equation (2)). After each outcome the estimated student parameters are adjusted. Using all data until time t we make a prediction (0 or 1) for the outcome at time $t + 1$. The accuracy is the number of correct predictions. We start with adding an outcome to all rules before starting prediction. The accuracy for different values of the student parameters ($\theta_{0,s}$ and η_s) is between 0.75 and 0.80.

3.2 Validation

To validate our model against reality, we use data from students working on solving problems in the domain of boolean algebra. As mentioned before there are 3 different groups of students. We estimate the parameters of the model on the basis of this data. The program produces the difficulty and discriminativity of the rules, and the starting competence and the learning parameter of the students as output.

As we know from our simulations, the parameter estimates for η_s and a_r are unreliable. We can therefore only draw conclusions for the parameter value estimates of b_r and $\theta_{0,s}$.

The estimates for starting competence can be compared to what we know about the different student groups. We expect the UU students to have most previous experience, and therefore the highest start competence. The HU business informatics students (HU-BI) are expected to have the lowest start competence, and the HU technical computer science students (HU-TCS) are expected to be in between. The means and median competences learned from data are: UU mean 1.27 and median 1.36, HU-TCS mean -0.19 and median -0.50, and HU-BI mean -0.90 and median -1.20. This confirms our hypothesis.

The domain experts rank the items by putting them in their perceived order of difficulty. In Figure 1 these rankings are shown together with the ranking calculated from the difficulties estimated from data. We observe that the expert rankings look similar to each other, and to the ranking learned from data. The main "surprise" is rule number 10: rewriting *something or true* to *true*.

We use Spearman's rank correlation on the different rankings. The correlations between expert rankings 1, 2 and 3, and the ranking estimated from data can be found in table 3.2. The estimated model parameters of difficulty and the difficulty rankings provided by experts are highly correlated. Also, the expert opinions do not deviate much more from each other than from the estimated ranking. This means that the model makes a similar estimated ranking of the difficulty of the rules as experts do. The model produces a this similar ranking, even though the amount of data is small.

Another important measure of the quality of the model is accuracy. We measure the accuracy by cross-validation. We set 1 student apart and estimate the rule parameters with the rest of the students. For the one student we set apart, we estimate the start competence and learning parameter

rule	Name	rewrite rule	diff (1)	diff (2)	diff (3)	diff (ML)
1.	Commutativity – and	p∧q ⇒ q∧p	3.5	12	3.5	2
2.	Commutativity – or	p∨q ⇒ q∨p	3.5	13	3.5	2
3.	Distribution – and over or	p∧(q∨r) ⇒ (p∧q)∨(p∧r)	18.5	22	19.5	20
4.	Distribution – or over and	p∨(q∧r) ⇒ (p∨q)∧(p∨r)	18.5	23	19.5	19
5.	Idempotency – and	p∧p ⇒ p	10.5	8	6	8
6.	Idempotency – or	p∨p ⇒ p	10.5	9	6	5
7.	T/F rules – tautology or	p∨¬p ⇒ p	12.5	11	12.5	12
8.	T/F rules – contradiction and	p∧¬p ⇒ p	12.5	10	12.5	9
9.	T/F rules – and True	p∧T ⇒ p	6.5	7	9.5	11
10.	T/F rules – or True	p∨T ⇒ T	8.5	4	9.5	22
11.	T/F rules – and False	p∧F ⇒ F	6.5	5	9.5	6
12.	T/F rules – or False	p∨F ⇒ p	8.5	6	9.5	10
13.	T/F rules – neg True	¬T ⇒ F	1.5	1	1.5	4
14.	T/F rules – neg False	¬F ⇒ T	1.5	2	1.5	2
15.	Double negation	¬¬p ⇒ p	5	3	6	7
16.	DeMorgan – on Or	¬(p∨q) ⇒ ¬p∧¬q	16.5	16	19.5	15
17.	DeMorgan – on And	¬(p∧q) ⇒ ¬p∨¬q	16.5	17	19.5	16
18.	Elimination – implication	p⟶q ⇒ ¬p∨q	22	15	14.5	17
19.	Elimination – equivalence	p⟷q ⇒ (p∧q)∨(¬p∧¬q)	23	14	17	18
20.	Absorption – outer or	(p∧q)∨p ⇒ p	20.5	20	22.5	21
21.	Absorption – outer and	(p∨q)∧p ⇒ p	20.5	21	22.5	23
22.	Tautology F impl	F⟶p ⇒ T	14.5	18	16	14
23.	Tautology A impl A	p⟶p ⇒ T	14.5	19	14.5	13

Table 1: The rewrite rules that can be applied while performing the DNF task, and the estimated difficulty rankings of these rules by experts 1 (Utrecht University of Applied Sciences), 2 (Utrecht University of Applied Sciences) and 3 (Utrecht University) and by maximum likelihood estimation.

	1	2	3	model
1	x	0.80	0.91	0.84
2	0.80	x	0.83	0.64
3	0.91	0.83	x	0.88
model	0.84	0.64	0.88	x

Table 2: The Spearman's rank correlations between experts 1, 2 and 3, and ranking based on the estimated model parameters.

after the first 25 outcomes. Then, for each outcome that follows, we predict whether it will be positive or negative, and determine whether our prediction is accurate. After the prediction we update the estimates for the start competence and learning parameters, before we predict the next outcome. There are 13 students with enough outcomes to warrant this prediction procedure. The average of the total number of outcomes per rule for these students is 6.1. The average accuracy of prediction for the 13 students we predicted the outcomes for is 0.78. For students with mostly correct outcomes the accuracy is not much higher than the percentage of correct answers. Around a proportion of correct outcomes of 0.5, the algorithm performs better. A typical example is student 12, with 52% correct answers, for who the accuracy was 0.71.

4. DISCUSSION

The purpose of this research is to create a student model for rule based e-tutoring systems. This model describes and predicts student behavior: whether or not a student applies a rule correctly. For this purpose, we have constructed a probability model based on 2PL IRT (equation 2).

We have used simulations to prove that the parameters of the model can be reliably learned; when we generate the data from a certain set of parameters, these parameters can be adequately recovered. For a small data set of 50 students, 25 rules, and 20 instances per rule, we show the parameters can be estimated with small error. For a data set of 15 students, 23 rules, and 8 instances per rule however, the estimates for the discriminativity of a rule, and the learning parameter of a student become unreliable. The accuracy of prediction for this small simulation however, is still between 0.75 and 0.80.

Using data from students working on rewriting logic expressions to DNF, we estimate the student and rule parameters from data obtained from 14 students, 23 rules and an average of 6.1 instances per rule. We compare the estimated difficulties of the rules to the rankings provided by domain experts, and show that the ranking based on the learning from data is as close to the rankings of the experts as the expert rankings are to each other, based on their Spearman's correlations. We show that the initial competence, $\theta_{0,s}$, of the students as learned from data, is what we expect based on what we know about the different groups of students who participated.

We cannot draw any conclusions about the result values of rule discriminativity (a_r) or student learning speed (η_s) because the real-life dataset is too small. We know from simulations however, that all model parameters are recoverable. To have reliable estimates for all parameters, the dataset should be obtained from around 50 students who apply each rule around 20 times.

We selected the domain of rewriting logic expressions because it is relatively well-known and often taught. In less well-known domains the model could be used to inform the teacher about the rule difficulty and, as hopefully further research will show, discriminativity. The model can also be used to report the competence to the students working on a rule based e-tutoring system.

We conclude that extended IRT is a promising model, which has the potential to provide more information to the users of rule based e-tutoring systems. More extensive tests with real-life data are required, but the results we obtained for the logic domain are promising.

5. REFERENCES

[1] Frank B. Baker and Seock-Ho Kim. *Item Response Theory - Parameter Estimation techniques (2nd ed.)*. Taylor and Francis group LLC, Boca Raton, FL, USA, 2004.

[2] Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of ITS-2006: the 8th International Conference on Intelligent Tutoring Systems*, pages 164–175, 2006.

[3] Bastiaan Heeren, Johan Jeuring, and Alex Gerdes. Specifying rewrite strategies for interactive exercises. *Mathematics in Computer Science*, 3(3):349–370, 2010.

[4] Jeff Johns, Sridhar Mahadevan, and Beverly Woolf. Estimating student proficiency using an item response theory model. In *Proceedings of ITS-2006: the 8th International Conference on Intelligent Tutoring Systems*, pages 473–480, 2006.